MN00058 6/28/2010

Protocol Manual

PRODUCT FAMILY: PTCR-20
PRODUCT NAME: PTCR-20 EMBEDDED CONTROLLER
PRODUCT MODELS: All Models that use PTCR-20 Control Circuit Board

MOOG QUICKSET
3650 WOODHEAD DRIVE
NORTHBROOK, IL 60062
TEL: 800 247-6563 • 847 498-0700 • FAX: 847 498-1258
WWW.QUICKSET.COM • E-MAIL: SALES@QUICKSET.COM

PTCR-20 Embedded Controller Protocol Rev D

(For use with the PTCR-20 PCB Rev D or above, firmware Rev 2.20 or above.)

Table of Contents

1. Pan & Tilt Communications Protocol Overview:

The controller PCB mounted internal to the QPT-20 (PTCR-20) will be addressable via a dedicated RS-422, RS-485 or RS-232 communications link provided by an attached remote server. The remote server will act as master and be responsible for initiating and maintaining communications with the PTCR. The PTCR will remain idle until the server sends a query/command. The PTCR will then parse the packet, check for integrity, and respond as required.

1.1 Autobaud:

An automatic baud detection procedure (autobaud) has been implemented. Supported baud rates are 9600bps, 14.4kbps, 19.2kbps, 28.8kbps, 38.4kbps, and 57.6kbps. Structure is 8 data bits, 1 stop bit, and no parity. Autobaud synchronization will only occur on power up and will not adapt "on the fly." The baud rate will be retained as long as power is applied to the PTCR. However, to change baud rates, the user may power the PTCR down, change host baud rate, then power the unit back up to redetect.

Normally, the user should make a practice of sending "keep alive" Get Status/Jog commands periodically if communications is lost in order to resync the PTCR to the host, even without the autobaud feature. With autobaud the host must be able to transmit up to 125-150 bytes of data (worst case) without a response expected in order for the PTCR to properly determine the data rate. This equates to about 15 Get Status/Jog commands. With a refresh rate of 120ms a maximum of about 2 seconds is required to determine baud rate. Once baud rate is determined the PTCR will begin responding to the Get Status/Jog commands and the host software may then move on to actual operation.

1.2 Storage of Coordinate Information:

The PTCR tracks absolute position using feedback from the precision potentiometers and/or encoders (resolvers) connected to the pan & tilt axes. This negates the need for a "homing" cycle at power-up. The unit inherently knows its absolute position. The feedback from the potentiometers is oversampled 16 times, then divided to produce a 13-bit position value ranging from 0-8191. The PTCR uses these values, referred to as "resolver units", directly to execute any automated movement commands. Resolver units are converted to angular values only for user display. Angular values supplied by the user are converted back to resolver units for both preset storage and automated movement. The advantage of using this method is increased accuracy and higher conversion speed during movement. If a user supplies an angular input for an automated move the PTCR converts it only once to a resolver unit value, then may directly compare this integer value to current resolver unit readings.

On continuous rotation units the pan position feedback is provided by a 13-bit per revolution incremental encoder. A running count is maintained, incremented, or decremented every 200us according to the current state of the encoder. This value is then stored in non-volatile RAM. The encoder also provides an index pulse once per revolution. When initially configured at the factory, this index point is equated to a resolver value relative to absolute center in pan and is stored in non-volatile EEPROM. The current running value stored in non-volatile RAM is updated frequently and will provide high accuracy at power up. However, if the platform is physically moved or bumped off position when powered down, the encoder changes will be missed and current position accuracy will be compromised. Whenever the index point is crossed during movement the encoder count will be automatically corrected using the stored index value. Optionally, the user may also reset to maximum accuracy by performing a homing cycle. See command 9EH. This command will automatically move the platform to the index point, realign the encoder count, and then return the platform to its original but corrected position.

Use of resolver units internally does have an effect on the structure of preset table entries and user-defined angular corrections. Presets are stored internally in absolute resolver units. The angular position for those presets is a calculated value derived from the conversion of resolver units to angles and any angular offsets defined by the user. Let us say, for example, that the user has not defined any angular offset. Therefore, the 0°/0° point for the platform will truly read 0°/0°. The user then defines a preset at +20° in pan and -10° in tilt. If the user moves to this preset, the angular reading returned will be +20°/-10°. The user then decides to redefine the 0°/0° position to return -10° in pan and 0° in tilt by entering a -10° pan correction. Though the physical 0°/0° position remains the same, the angular reading for this position will now be returned as -10°/0°. If the user then moves to the former +20°/-10° preset the platform will move to the same preset position but the angle returned will be +10°/-10°.

There is an important rational behind this coordinate system. Many will use the angular readings of the platform to observe the relative position of different targets. For example, the user may find that one object resides at -30° pan/-10° tilt relative to a true 0°/0° point. A second object resides at +25° pan/-20° tilt relative to the 0°/0° point. The user moves back to the first object and uses the "Align Angles to Center" command. This will introduce a pan angle correction of +30° and a tilt correction of +10°, adjusting the returned angular reading for the first object to 0°/0°. If the user now moves to the second object the angular reading will be +55° pan/-10° tilt, the actual angular distance from the first object.

Therefore, presets will always reflect known physical positions of the platform. No matter how the pan and tilt angles are shifted, a preset position will remain valid and correct and a command to move to that position will still return the platform to it. Introducing angular corrections will do nothing more than change the <u>returned angular position</u> but will not change the actual physical position of a preset. Additionally, software limits will also show this shift. Since a software limit is designed to prohibit movement beyond a specific point relative to center, the actual position of the software limit will not change when an angular offset is set. Only the value returned will change.

1.3 Camera and Lens Control:

A non-volatile parameter is set and stored for identifying the lens type. Parameters are also stored for determining whether a camera has serial remote control capabilities and, if so, what the baud rate, number of data bits, and parity are. At start-up the main microcontroller reads these values from its internal EEPROM and configures the lens drivers and camera serial port as required. This determines how the slave interprets data and formats it for the camera and lens.

The microcontroller used for the PTCR contains two PWM drive sections; one for platform motor control and another for zoom/focus motor control. Motor driven lenses are divided into two types, 12VDC bipolar and 6VDC crosspolar.

In the bipolar drive configuration each motor (one for zoom, one for focus) has two discrete power leads (4-wire.) 0V is supplied to the two motor leads when no movement is required. Lead A is made more positive than lead B to move in one direction. This is affected by holding lead B at ground and supplying a PWM signal of increasing high duty cycle producing an output for lead A climbing from a minimum of 0V to a maximum of +12VDC. Lead A is made more negative than lead B to move in the opposite direction. This is affected by holding lead B at +12VDC and supplying a PWM signal of increasing low duty cycle producing an output falling from a maximum of +12VDC to a minimum of 0V.

In the crosspolar drive configuration each motor has one discrete power lead and the remaining leads tied together to a common lead (3-wire.) As with bipolar drive, all three leads are supplied with 0V when no movement is required. However, when there is a requirement for movement, all three leads are initially supplied with +6VDC. The effective difference between the leads is still 0V so no movement occurs. The common lead is held at +6VDC and the discrete lead for each motor is shifted from 0V to +12VDC to affect both a change in speed and, as the +6VDC common point is crossed, a change in direction.

Lenses are fitted with potentiometers for positional feedback. The potentiometer output is connected to the microcontroller ADC's and is converted to a binary value indicating current position. This value is then returned to the host. The host may send a request to move the zoom and focus motors to either an undetermined final position (jog) or a predetermined final position (move to.) These predetermined final positions are stored as part of the pan and tilt's preset table, allowing the system to move to specific pan and tilt coordinates while also adjusting zoom and focus for the position.

Camera control is more difficult due to the numerous control signals available for each camera type. Therefore, a special variable-length command 62H is used to wrap a complete camera command for transfer to the camera. The user should build a complete camera command string. It should then be prepended and appended with the appropriate PTCR control bytes for transfer to the main microcontroller. The main microcontroller will then strip the intact camera command out of the packet and transfer it to the slave microcontroller where it will then be sent to the camera. See command 62H, "Command Camera", below for more detail.

The PTCR can also return received camera data through the “Get Status/Jog” command and response. The PTCR indicates that a camera string is available by changing the “cam count” byte from 0 to the number of bytes returned. When the host sees this change the indicated number of bytes should be removed from the response and stored for camera response parsing. The PTCR will return the camera's data string verbatim, including all control characters, checksums, etc. See command 31H for more information.

1.4 General Communications Structure:

The general structure of the host-to-PTCR communications protocol is as follows:

Data	Format	Bytes	7	6	5	4	3	2	1	0
STX	02H	1								
Identity	xxH	1	1-99 (01H-63H) for RS-485 identity or 00 for dedicated/broadcast							
Command	xxH	1								
Data										
▼										
Data										
LRC	xxH	1								
ETX	03H	1								

A transfer from the host to the PTCR-20 will always start with a unique STX (Start-of-Text) 02H character followed by an identity address for the connected unit. The command number to execute and any command data will follow. The packet will then be completed by sending the LRC (Longitudinal Redundancy Checksum) and the ETX (End-of-Text) 03H character.

Data	Format	Bytes	7	6	5	4	3	2	1	0
ACK	06H	1								
Identity	xxH	1	1-99 (01H-63H) for RS-485 identity or 00 for dedicated/broadcast							
Command	xxH	1								
Data										
▼										
Data										
LRC	xxH	1								
ETX	03H	1								

The response from the PTCR-20 is very similar. It will always start with a unique ACK (Acknowledge) 06H character followed by an identity address for the connected unit. The unit will echo the command number for confirmation of the received command and include any data associated with the command number. The packet will then be completed by sending the LRC (Longitudinal Redundancy Checksum) and the ETX (End-Of-Text) 03H character.

When operating in RS-485 daisy-chain mode the identity address is used to determine which of the connected pan and tilt units is being commanded and should respond. All others will ignore the command and remain quiet. This address is set using Command 9FH and can range from 00-99. Address 00 is reserved for dedicated RS-232 and RS-422 mode and for broadcast mode. When the 00 address is used it will be internally ignored and any attached PTCR will respond. The current PTCR identity will be returned as part of the standard responses. If the user needs to change the identity of a unit the 9FH identity command should be sent using the current identity as the identity address and the new identity as the data byte. The PTCR will return the current identity as its identity address and the new identity as its response. **From that point forward, the new identity will take effect.** If the user forgets the current identity it can always be set and retrieved using an identity address of 00. **However, all other platforms in the network <u>must</u> be disconnected before issuing ANY commands using address 00. Otherwise, all units will react to the command and seize the host receive line.**

The PTCR will always echo the last received command number back to the remote in its response. Remote software can be configured to accept this echo as a confirmation that the command was actually received and will be responded to. In the command breakdowns that follow, the transmission from remote to PTCR will always be followed by the response from PTCR to remote. The command listings will show only the command number and pertinent data. **However, all transfers require and will return the control characters in the sequence listed above.**

1.5 Control Characters:

The following are definitions for control characters used in data transfer.

Char	Description	Sent By	Value
STX	Start of Text	Remote	02H
ETX	End of Text	Remote/PTCR	03H
ACK	Acknowledge	PTCR	06H
NAK	Not Acknowledge	PTCR	15H

1.6 Calculating the Checksum:

The checksum used for data transfer is a longitudinal redundancy check or LRC. It is calculated by XOR'ing bytes starting with the command number and ending with the last data byte. The ACK/NAK/STX and ETX are **<u>not</u>**

included in the LRC. The easiest method of calculating and comparing is to XOR all data bytes, then XOR the result with the LRC checksum. The result should be 0 (zero).

1.7 Passing Data that Matches Control Character Values (ESC/bit-7 Set):

When passing full 8-bit bitsets it is possible that a value may match a control character (ACK/NAK/STX/ETX.) Therefore, the protocol needs some method of distinguishing these values from control characters. The method used is the insertion of an ESC character prior to transmitting the conflicting data byte and the setting of Bit-7 of the conflicting byte. Since we must also be able to distinguish the ESC value of 1BH we will perform the same operation on ESC's.

Example: Data to send = 02H Data sent = 1BH 82H
Data to send = 1BH Data sent = 1BH 9BH

This insertion should be performed on **any byte that is not a control character, including the LRC.** Note that this procedure should be performed immediately prior to transmission and the companion decoding should be performed prior to checksum calculation after reception. These insertions are not included in the LRC calculations. The entire receive buffer should be scanned prior to LRC check and parsed for any occurrences of ESC. The ESC should be tossed, the following byte should have Bit-7 cleared, then the buffer should be shifted down. The buffer will then be ready for LRC calculation and data parsing.

The inclusion of ESC sequences provides two distinct advantages. First, the user is assured that any reception of an ACK or ETX is valid. Unless an error occurs, these control characters will not show up as data bytes in the packet. Therefore, it is perfectly valid to cue the start of reception on any ACK character. It is also valid to cue the end of reception and begin parsing data on the reception of an ETX. Since the ACK and ETX are not used in the calculation of the LRC, the user can simply use them as starting and stopping cues. Secondly, unlike the original implementation of IBM bisync on which this protocol was based, the user is allowed to pass full 8-bit binary values. Code snippets are provided at the end of this document demonstrating different approaches in C for implementing LRC calculation, LRC checking, and ESC insertion and removal.

1.8 Passing Integer Values:

As noted in the protocol command descriptions, some values sent between the remote and PTCR units are integer values. These integer values should be passed and received as 16-bit signed two's-complement little endian integers simply split between two bytes. The first byte should represent the LSB of the integer with the second byte containing the MSB of the integer. Negative values are represented as the two's-complement of the positive value. For example:

Integer Value	Integer in Hex	First Byte	Second Byte
32767	7FFFH	FFH	7FH
2	0002H	02H	00H
1	0001H	01H	00H
0	0000H	00H	00H
-1	FFFFH	FFH	FFH
-2	FFFEH	FEH	FFH
-32768	8000H	00H	80H

1.9 The Programmer's Responsibility for Input Range and Format

The microcontroller used for the PTCR has a relatively small amount of code space. Extensive range and format checking of all user input would seriously task the processor and limit the amount of space left for executable procedures. The listings for each command in this document present the allowable range for each byte or integer of data. The user/programmer is responsible for providing inputs that are properly formatted and within the specified numeric range for each command. However, **absolute coordinates, preset numbers and static tour step locations** are checked and flagged if out of range or otherwise invalid.

1.10 Status Definitions:

Bits are active high, i.e., "set" or "1" indicates the condition exists. (S)oft faults are self-healing. (H)ard faults require a RESET (RES) command to clear.

Sym	Type	Name	Description
xHL	S Fault	Hard Limit	An axis hard limit has been reached.
xSL	S Fault	Soft Limit	An axis soft limit has been reached.
TO	H Fault	Timeout	A commanded axis is not moved within the prescribed timeframe.
DE	H Fault	Direction Error	A commanded axis has moved in the wrong direction.
OL	H Fault	Current Overload	A commanded axis has tripped the current overload.
xRF	S Fault	Resolver Fault	A resolver is disconnected or not operating properly.
xxxM	Stat	Moving	The commanded axis is currently moving.
OSLR	Stat	Override Return	The controller is in soft limit override.
DES	Stat	Destination	The coordinates returned are destination coordinates, not current.
EXEC	Stat	Executing	The PTCR is executing a remote initiated command.
ENC	Stat	Encoder Installed	Platform is encoder-based and pan soft/hard limits are ignored.

PTCR Command Set:

This section includes information on the actual commands used to control and monitor the pan & tilt unit and attached camera.

1.11 Numeric Command List:

Cmd	Name	Cmd	Name
31H	Get Status/Jog	67H	Get/Set Camera/Lens Type
32H	Move To Preset	68H	Get/Set Auto Focus Mode
33H	Move To Entered Coordinates		
34H	Move To Delta Coordinates	80H	Set Pan & Tilt Angle Correction
35H	Move To Absolute 0/0	81H	Get/Set Soft Limit
36H	Move To Home	82H	Align Angles To Center
37H	Start Preset Tour	83H	Align Angles To Coordinates
		84H	Clear Angle Correction
40H	Retrieve Preset Table Entry	85H	Get Pan & Tilt Angle Correction
41H	Save Coordinates As Preset Table Entry	86H	Get/Set Ramp Down Points in RU's
42H	Save Current Position As Preset Table Entry		
43H	Add Current Z/F To Preset Table Entry	90H	Get Center Position in RU's
		91H	Set Center Position
50H	Flush Preset Tour	92H	Get Minimum Speeds
51H	Query Preset Tour	93H	Set Minimum Speeds
52H	Append To Preset Tour	94H	Initialize Preset Table to 0/0
53H	Insert Into Preset Tour	95H	Get/Set Motor and Resolver Direction
54H	Delete From Preset Tour	96H	Get/Set Communication Timeout
55H	Replace In Preset Tour	98H	Get Maximum Speeds
56H	Get Tour Size	99H	Set Maximum Speeds
		9AH	Get Firmware Revision
60H	Get/Set Camera Comm Parameters	9BH	Get Clock
61H	Get/Set Lens Parameters	9CH	Set Clock
62H	Command Selected Camera	9DH	Initial Encoder Align
65H	Get/Set Camera Timeouts	9EH	Perform Homing Cycle

66H	Get/Set Aux Control Outputs	9FH	Get/Set Identity Address

“Get Status/Jog” & "Keep Alive" Command

This command can be used as a standard "Keep Alive" from the remote unit to continuously gather coordinate data. This command should be the one sent when no other command is required. This will keep the remote computer advised of current position and any faults that may exist. This will also confirm to the PTCR that the remote computer is connected and properly communicating.

The jog bytes contain both a direction and a speed value. This allows proportional, simultaneous jog control of both axes. Holding the speed value for an axis at 0 will prohibit jogging that axis. Note that any jog command that includes a speed value other than 0 will automatically stop any automated command. Therefore, initiating a jog can be used to terminate an automated move, including a preset tour.

The two auxiliary commands provide focus and zoom jog control. These commands embed both speed and direction for zoom and focus. Zoom and focus can be adjusted independently of pan & tilt jog.

The "STOP" bit can also be used to stop all motors during execution of an automated movement command. For example, if a "MOVE TO" is to be executed the remote would send the "MOVE TO" command once, then return to the "GET STATUS/JOG" command to update position display. If the user wishes to stop the "MOVE TO" command from the remote the "STOP" bit should be set then, after confirmation of reception, cleared. This will cause the PTCR to stop the motors, fall out of the "MOVE TO" procedure, and return to a state waiting for the next command.

The "RES" or reset bit is used to clear latching (hard) faults. These include motor directional errors (DE) and timeouts (TO). A timeout fault will be set if an axis fails to move within 1 second. This may be the result of a stalled motor, an overloaded platform, or a physical obstruction prohibiting motor movement. A directional error fault will be set if an axis is detected as moving in the wrong direction. This may be the result of improper motor wiring. **Note that the DE and TO faults will only occur during automated moves.** The user should observe the angular readings during jog to confirm motors are moving the proper direction and are not stalled.

The "OSL" bit allows overriding soft limits during jog. **This bit should only be set when initially setting up the soft limits.** Its current setting will be returned in the OSLR bit.

Setting the "RU" bit forces the PTCR to return any status response coordinate data as resolver units rather than angles. This function can be helpful when initially aligning potentiometers to the platform. Output range is 0-4095.

Data	Format	Bytes	7	6	5	4	3	2	1	0
Cmd Num	31H	1								
Cmd	Bitset	1	0	0	0	0	RU	OSL	STOP	RES
Pan Jog Cmd	Bitset	1	Pan Speed (0-127)							Dir[1]
Tilt Jog Cmd	Bitset	1	Tilt Speed (0-127)							Dir[2]
Zoom Jog	Bitset	1	Zoom Speed (0-127)							Dir[3]
Focus Jog	Bitset	1	Focus Speed (0-127)							Dir[4]

[1] 1 = CW/0 = CCW 0 Speed = No Movement
[2] 1 = UP/0 = DWN 0 Speed = No Movement
[3] 1 = Zoom Out/0 = Zoom In 0 Speed = No Movement
[4] 1 = Focus Out/0 = Focus In 0 Speed = No Movement

Response	Format	Bytes	7	6	5	4	3	2	1	0
Cmd Num	31H	1								
PAN Coord	Int	2	PAN = -3600 to +3600 = -360.0° to +360.0°							
TILT Coord	Int	2	TILT = -1800 to +1800 = -180.0° to +180.0°							
PAN Status	Bitset	1	CWSL	CCWSL	CWHL	CCWHL	TO	DE	OL	PRF[2]

TILT Status	Bitset	1	USL	DSL	UHL	DHL	TO	DE	OL	TRF[2]
Gen Status	Bitset	1	ENC	EXEC	DES[1]	OSLR	CWM	CCWM	UPM	DWNM
Zoom Coord	Byte	1	1-255							
Focus Coord	Byte	1	1-255							
Cam Count	Byte	1	0-24 for number of bytes of camera data to follow							
Camera Data	Bytes	0-24	Camera return string in native format (only present if Cam Count > 0)							

[1]DES bit is clear if coordinates are current, set if coordinates are the destination of a MOVE TO command.
[2]The PRF and TRF resolver fault bits are only implemented in controllers fitted with synchro-resolvers.

1.12 Automated "Move To" Commands:

Any “Move To” command should only be repeated until an acknowledgement has been received from the PTCR (echo of the command number). The remote should then revert back to the standard 31H "Get Status/Jog" query. The PTCR will set the EXEC (executing) bit in the general status bitset to indicate that the command is being carried out. This bit will clear once the move has been completed.

The PTCR response to any automated "Move To" command will be identical to the standard status response with one exception. The response will echo the destination coordinates either as entered or retrieved from the preset table rather than the current coordinates. The setting of the destination bit DES, bit-5 of general status will indicate this. Status will then default back to current coordinates once the "Get Status/Jog" command/response resumes and the DES bit clears. The user may cue on the DES bit in order to fill a "Moving To" window with the destination coordinates.

If the PTCR is detecting hard faults that will prohibit executing a “Move To” command it will echo the current position as the destination coordinates. This should act as a reminder for the user to check the fault status.

Movement will start once the PTCR has parsed the “Move To" coordinates. The setting of the appropriate axis MOVE bits in the status response will indicate this. As each axis arrives on station the respective MOVE bit will be cleared. The remote may assume the move has been completed when all MOVE bits and the EXEC bit have cleared. If a fault occurs on **any** axis **all** motors will stop. The fault will be set and all MOVE bits and the EXEC bit will be cleared.

Setting the STOP bit or setting jog speed to a value other than 0 in any following "Get Status/Jog" command will immediately terminate any automated "Move To" operation. Sending any command other than the 31H "Get Status/Jog" command during an automated move will also terminate the automated "Move To" command.

1.12.1 “Move To Preset” Command:

The PTCR can retain up to 32 position entries in a non-volatile table that are frequently used by the operator. This command is used to move the platform to a preset position defined in this preset table. Further information on setting up the preset table is provided under command 40H below. If a preset number greater than 31 is entered no move will occur.

Data	Format	Bytes	
Command	32H	1	
Preset	xxH	1	Preset Number 0-31

Response	Format	Bytes	7	6	5	4	3	2	1	0
Command	32H	1								
PAN Coord	Int	2	PAN = -3600 to +3600 = -360.0° to +360.0°							
TILT Coord	Int	2	TILT = -1800 to +1800 = -180.0° to +180.0°							
PAN Status	Bitset	1	CWSL	CCWSL	CWHL	CCWHL	TO	DE	OL	PRF
TILT Status	Bitset	1	USL	DSL	UHL	DHL	TO	DE	OL	TRF
Gen Status	Bitset	1	ENC	EXEC	1	OSLR	CWM	CCWM	UPM	DWNM

Zoom Coord	Byte	1	1-255
Focus Coord	Byte	1	1-255

1.12.2 "Move To Entered Coordinate" Command:

This command is used to move the platform to a specific set of manually entered coordinates. The coordinate must consist of the desired position to 1/10th degree multiplied by 10, i.e., +90.0° should be sent as 900. The user can force an axis to remain in position by sending its current position back to the PTCR. However, the coordinate value 9999 (+999.9°) can also be sent in order to prohibit movement of a specific axis. For example, if the user wishes to only move PAN, send 9999 (+999.9°) as the "Move To" coordinate for TILT and the TILT axis will remain stationary.

The PTCR will perform a range check for the input coordinates and abort the move if a coordinate is out of range. Allowable pan range is defined as -180.0° + pan angle offset to +180.0° + pan angle offset. Allowable tilt range is -90.0° + tilt angle offset to +90.0° + tilt angle offset. For an angle offset of 0°/0°, the range would be -180.0°/+180.0° and -90.0°/+90.0°. If a pan angle correction of +180.0° is entered, the allowable pan angle range would shift to 0.0° to 360.0°. If a tilt angle correction of -20.0° is entered, the allowable tilt angle range would shift to -110.0° to 70.0°.

Data	Format	Bytes	
Command	33H	1	
PAN Coord	Int	2	PAN = -3600 to +3600 = -360.0° to +360.0° or 9999 for no move
TILT Coord	Int	2	TILT = -1800 to +1800 = -180.0° to +180.0° or 9999 for no move

Response	Format	Bytes	7	6	5	4	3	2	1	0
Command	33H	1								
PAN Coord	Int	2	PAN = -3600 to +3600 = -360.0° to +360.0°							
TILT Coord	Int	2	TILT = -1800 to +1800 = -180.0° to +180.0°							
PAN Status	Bitset	1	CWSL	CCWSL	CWHL	CCWHL	TO	DE	OL	PRF
TILT Status	Bitset	1	USL	DSL	UHL	DHL	TO	DE	OL	TRF
Gen Status	Bitset	1	ENC	EXEC	1	OSLR	CWM	CCWM	UPM	DWNM
Zoom Coord	Byte	1	1-255 (01H-FFH)							
Focus Coord	Byte	1	1-255 (01H-FFH)							

1.12.3 "Move To Delta Coordinates" Command:

As opposed to moving to specific coordinates, the "Move To Delta Coordinate" command allows the user to move the platform a specific angular distance from the current position. The coordinate must consist of the desired position to 1/10th degree multiplied by 10, i.e., -20.0° should be sent as -200. The user can force an axis to remain stationary by sending 0 to the PTCR for that axis.

Data	Format	Bytes	
Command	34H	1	
PAN Coord	Int	2	PAN = -3600 to +3600 = -360.0° to +360.0°
TILT Coord	Int	2	TILT = -1800 to +1800 = -180.0° to +180.0°

Response	Format	Bytes	7	6	5	4	3	2	1	0
Command	34H	1								
PAN Coord	Int	2	PAN = -3600 to +3600 = -360.0° to +360.0°							
TILT Coord	Int	2	TILT = -1800 to +1800 = -180.0° to +180.0°							
PAN Status	Bitset	1	CWSL	CCWSL	CWHL	CCWHL	TO	DE	OL	PRF
TILT Status	Bitset	1	USL	DSL	UHL	DHL	TO	DE	OL	TRF
Gen Status	Bitset	1	ENC	EXEC	1	OSLR	CWM	CCWM	UPM	DWNM
Zoom Coord	Byte	1	1-255 (01H-FFH)							

Focus Coord	Byte	1	1-255 (01H-FFH)

1.12.4 "Move To Absolute 0/0" Command:

The Pan & Tilt unit potentiometer resolvers are initially aligned with the platform centered and level. This command will return the platform to that stored center position. This is a convenient method for returning the platform to factory center for maintenance or hard limit switch alignment.

Data	Format	Bytes
Command	35H	1

Response	Format	Bytes	7	6	5	4	3	2	1	0
Command	35H	1								
PAN Coord	Int	2	PAN = -3600 to +3600 = -360.0° to +360.0°							
TILT Coord	Int	2	TILT = -1800 to +1800 = -180.0° to +180.0°							
PAN Status	Bitset	1	CWSL	CCWSL	CWHL	CCWHL	TO	DE	OL	PRF
TILT Status	Bitset	1	USL	DSL	UHL	DHL	TO	DE	OL	TRF
Gen Status	Bitset	1	ENC	EXEC	1	OSLR	CWM	CCWM	UPM	DWNM
Zoom Coord	Byte	1	1-255 (01H-FFH)							
Focus Coord	Byte	1	1-255 (01H-FFH)							

1.12.5 "Move To Home" Command:

A special preset position number 31, referred to as "Home", may be entered and stored by the PTCR. This command requires no preset number or coordinate input and will always return the pan and tilt unit to this "Home" position.

Data	Format	Bytes
Command	36H	1

Response	Format	Bytes	7	6	5	4	3	2	1	0
Command	36H	1								
PAN Coord	Int	2	PAN = -3600 to +3600 = -360.0° to +360.0°							
TILT Coord	Int	2	TILT = -1800 to +1800 = -180.0° to +180.0°							
PAN Status	Bitset	1	CWSL	CCWSL	CWHL	CCWHL	TO	DE	OL	PRF
TILT Status	Bitset	1	USL	DSL	UHL	DHL	TO	DE	OL	TRF
Gen Status	Bitset	1	ENC	EXEC	1	OSLR	CWM	CCWM	UPM	DWNM
Zoom Coord	Byte	1	1-255 (01H-FFH)							
Focus Coord	Byte	1	1-255 (01H-FFH)							

1.13 The Preset Table:

The PTCR can retain up to 32 (0-31) preset positions in non-volatile memory that are frequently used by the operator. The user can store, retrieve, or move to these coordinates by modifying and using the preset table as outlined below. Preset zoom and focus positions can also be stored.

A special preset, referred to as "Home" position, can be directly driven to without referencing a preset number using command 36H. It should be saved at preset position 31. This position can also be assigned as a generic preset and can be included in a static preset tour.

Any command to store or retrieve a preset entry will echo back the preset's coordinates. If a preset number greater than 31 is entered the preset number will be echoed as FFH and the remainder of the data will be 0's.

1.13.1 "Retrieve Preset Table Entry" Command:

The operator may retrieve the stored coordinate position and zoom/focus byte values for any preset.

Data	Format	Bytes	
Command	40H	1	
Preset Num	xxH	1	0-31 (0-1FH)

Response	Format	Bytes	
Command	40H	1	
Preset Num	xxH	1	0-31 (0-1FH) or FFH if preset out of range
Preset Pan	Int	2	PAN = -3600 to +3600 = -360.0° to +360.0°
Preset Tilt	Int	2	TILT = -1800 to +1800 = -180.0° to +180.0°
Zoom Coord	xxH	1	1-255 (01H-FFH)
Focus Coord	xxH	1	1-255 (01H-FFH)

1.13.2 "Save Coordinates As Preset Table Entry" Command:

This command allows the user to load a specific set of coordinates and zoom/focus bytes to the preset table. The PTCR will perform a range check for the input coordinates and abort saving if a coordinate is out of range. Allowable pan range is defined as -180.0° + pan angle offset to +180.0° + pan angle offset. Allowable tilt range is -90.0° + tilt angle offset to +90.0° + tilt angle offset. For an angle offset of 0°/0°, the range would be -180.0°/+180.0° and -90.0°/+90.0°. If a pan angle offset of +180.0° is entered, the allowable pan angle range would shift to 0.0° to 360.0°. If a tilt angle offset of -20.0° is entered, the allowable tilt angle range would shift to -110.0° to 70.0°. The user may save coordinates that exceed both soft and hard limits.

Data	Format	Bytes	
Command	41H	1	
Preset Num	xxH	1	0-31 (0-1FH)
Preset Pan	Int	2	PAN = -3600 to +3600 = -360.0° to +360.0°
Preset Tilt	Int	2	TILT = -1800 to +1800 = -180.0° to +180.0°
Zoom Coord	xxH	1	1-255 (01H-FFH)
Focus Coord	xxH	1	1-255 (01H-FFH)

Response	Format	Bytes	
Command	41H	1	
Preset Num	xxH	1	0-31 (0-1FH) or FFH if preset out of range
Preset Pan	Int	2	PAN = -3600 to +3600 = -360.0° to +360.0°
Preset Tilt	Int	2	TILT = -1800 to +1800 = -180.0° to +180.0°
Zoom Coord	xxH	1	1-255 (01H-FFH)
Focus Coord	xxH	1	1-255 (01H-FFH)

1.13.3 "Save Current Position As Preset Table Entry" Command:

This command allows the user to store the platform's current position and zoom and focus positions as a preset table entry.

Data	Format	Bytes	
Command	42H	1	
Preset Num	xxH	1	0-31 (0-1FH)

Response	Format	Bytes	
Command	42H	1	
Preset Num	xxH	1	0-31 (0-1FH) or FFH if preset out of range

Preset Pan	Int	2	PAN = -3600 to +3600 = -360.0° to +360.0°
Preset Tilt	Int	2	TILT = -1800 to +1800 = -180.0° to +180.0°
Zoom Coord	xxH	1	1-255 (1-FFH)
Focus Coord	xxH	1	1-255 (1-FFH)

1.13.4 "Add Current Z/F To Preset Table Entry" Command:

With Command 41H the user can provide and store an absolute position for pan and tilt destination angles in a preset table entry. However, the zoom and focus entries for an absolute preset are relative. Using this command, the user can append the current zoom and focus positions (usually set by jogging) to an absolute preset entry.

Data	Format	Bytes	
Command	43H	1	
Preset Num	xxH	1	0-31 (0-1FH)

Response	Format	Bytes	
Command	43H	1	
Preset Num	xxH	1	0-31 (0-1FH) or FFH if preset out of range
Preset Pan	Int	2	PAN = -3600 to +3600 = -360.0° to +360.0°
Preset Tilt	Int	2	TILT = -1800 to +1800 = -180.0° to +180.0°
Zoom Coord	xxH	1	1-255 (1-FFH)
Focus Coord	xxH	1	1-255 (1-FFH)

1.14 The Preset Tour:

The PTCR can hold three 63-step (0-62) preset tours. Preset tours are built only from assigned presets and allow the pan and tilt unit to sequentially move to a preset in the tour, wait a defined period of time, move to the next preset in the tour, wait a defined period of time, etc. Tours will be continuously executed until a "STOP" or jog command is received or a command other than a status query is received. **If a fault occurs or a soft or hard limit is reached the tour will stop executing.**

Tours are built by first flushing the existing tour. This will reset the tour pointer to 0. The user then sequentially adds preset numbers (0-31) and wait times (0-99 secs) to each stop in the tour. Once built, the tour can be started using the "Start Preset Tour" command, selecting which tour to execute.

The user also has the capability to edit the preset tour using the "Append To Preset Tour", "Insert Into Preset Tour", "Delete From Preset Tour", and "Replace In Preset Tour" commands. The first command simply adds a preset to the end of the tour. The second allows the user to insert a preset into the tour while retaining the presets that follow. The third allows the user to remove a preset while retaining the presets that follow. In both cases, the presets that follow will be shifted up or down as required to keep the tour complete. The fourth command allows the user to replace a tour entry without disturbing the remaining tour entries.

1.14.1 “Start Preset Tour" Command:

This command allows the user to start either one of the three static preset tours. If the tour is empty FFH will be returned. See each type below for further tour information.

Data	Format	Bytes	
Command	37H	1	
Tour Num	xxH	1	0-2

Response	Format	Bytes	7	6	5	4	3	2	1	0
Command	37H	1								
Tour Num	xxH	1	0-2 or FFH if selected tour is empty							

1.14.2 “Flush Preset Tour” Command:

The operator may completely clear a tour and ready it for building by using this command.

Data	Format	Bytes	
Command	50H	1	
Tour Num	xxH	1	0-2

Response	Format	Bytes	
Command	50H	1	
Tour Num	xxH	1	0-2

1.14.3 “Query Preset Tour” Command:

The operator may examine the sequential steps of a tour by using this command. If the response returns FFH for the step number, the requested step does not exist in the tour.

Data	Format	Bytes	
Command	51H	1	
Tour Num	xxH	1	0-2
Step Num	xxH	1	0-62 (0-3EH)

Response	Format	Bytes		
Command	51H	1		
Tour Num	xxH	1	0-2	
Step Num	xxH	1	0-62 (0-3EH) or FFH for all if the step does not exist	
Preset Num	xxH	1	0-31 (0-1FH)	
Wait Time	xxH	1	0	0-99 (0-63H) seconds

1.14.4 “Append To Preset Tour” Command:

The operator may append a preset to the tour by using this command. The step number returned by the response represents the tour position where the preset was saved. If FFH is returned the tour is already full.

Data	Format	Bytes		
Command	52H	1		
Tour Num	xxH	1	0-2	
Preset Num	xxH	1	0-31 (0-1FH)	
Wait Time	xxH	1	0	0-99 (0-63H) seconds

Response	Format	Bytes

Command	52H	1		
Tour Num	xxH	1	0-2	
Step Num	xxH	1	0-62 (0-3EH) or FFH for all if the current tour is full	
Preset Num	xxH	1	0-31 (0-1FH)	
Wait Time	xxH	1	0	0-99 (0-63H) seconds

1.14.5 "Insert Into Preset Tour" Command:

The operator may insert a preset into the tour by using this command. Any steps that follow the insertion will be moved out one step. If a step number of FFH is returned the tour is full and the preset was not accepted. If the step number returned is less than the step number sent the step did not originally exist and the new entry was appended to the tour.

Data	Format	Bytes		
Command	53H	1		
Tour Num	xxH	1	0-2	
Step Num	xxH	1	0-62 (0-3EH)	
Preset Num	xxH	1	0-31 (0-1FH)	
Wait Time	xxH	1	0	0-99 (0-63H) seconds

Response	Format	Bytes		
Command	53H	1		
Tour Num	xxH	1	0-2	
Step Num	xxH	1	0-62 (0-3EH) or FFH for all if the current tour is full	
Preset Num	xxH	1	0-31 (0-1FH)	
Wait Time	xxH	1	0	0-99 (0-63H) seconds

1.14.6 "Delete From Preset Tour" Command:

The operator may delete a preset in the tour by using this command. Any steps that follow the deletion will be moved down one step. If a step number of FFH is returned the tour did not contain the step to be deleted.

Data	Format	Bytes	
Command	54H	1	
Tour Num	xxH	1	0-2
Step Num	xxH	1	0-62 (0-3EH)

Response	Format	Bytes		
Command	54H	1		
Tour Num	xxH	1	0-2	
Step Num	xxH	1	0-62 (0-3EH) or FFH for all if the step does not exist	
Preset Num	xxH	1	0-31 (0-1FH)	
Wait Time	xxH	1	0	0-99 (0-63H) seconds

1.14.7 "Replace In Preset Tour" Command:

The operator may replace a preset in the tour by using this command. All other existing steps will be unaltered. If a step number of FFH is returned the tour did not contain the step to be replaced.

Data	Format	Bytes		
Command	55H	1		
Tour Num	xxH	1	0-2	
Step Num	xxH	1	0-62 (0-3EH)	
Preset Num	xxH	1	0-31 (0-1FH)	
Wait Time	xxH	1	0	0-99 (0-63H) seconds

Response	Format	Bytes		
Command	55H	1		
Tour Num	xxH	1	0-2	
Step Num	xxH	1	0-62 (0-3EH) or FFH for all if the step does not exist	
Preset Num	xxH	1	0-31 (0-1FH)	
Wait Time	xxH	1	0	0-99 (0-63H) seconds

1.14.8 "Get Tour Size" Command:

The operator can query a tour to find out the number of steps stored. If the tour is corrupt this procedure will return FFH or 255. This is a good method of determining if the tour is corrupt before attempting to modify it.

Data	Format	Bytes	
Command	56H	1	
Tour Num	xxH	1	0-2

Response	Format	Bytes	
Command	56H	1	
Tour Num	xxH	1	0-2
Step Count	xxH	1	0-62 (0-3EH) or FFH if the tour is corrupt

1.15 Camera/Lens Parameters and Control

The PTCR will act as a transparent conduit for camera commands. Any command destined for the camera should be completely built including any control characters and checksums. It should then be placed inside a pan & tilt command wrapper (see command 62H.) Up to 24 bytes may be sent in one string to the camera. The PTCR will be made aware of the camera's required serial parameters by the setting of its own non-volatile parameters (see command 60H.) The PTCR will use this data to configure a UART and send the string to the camera.

Normally, the PTCR will respond to a command 31H status query with standard status data and a camera byte count of 0. When a serial byte is received from the camera a timer will be started and the PTCR will wait for additional bytes to arrive (see command 65H.) As each byte is received it will be placed in a queue. Once the timer expires the PTCR will tack the byte count and returned camera data onto the end of the next status response. Therefore, if a camera byte count greater than 0 is received the application should remove the camera bytes and use them as required.

Though a bit convoluted, this allows the PTCR to be transparent, negating the need for code changes to accommodate different camera types. This will also allow cameras that only provide dedicated RS-232 or RS-422 communications to participate in an RS-485 party line environment. However, the onus of building proper camera commands and parsing received data resides with the programmer.

1.15.1 "Get/Set Camera Comm Parameters" Command:

These parameters define the serial communications parameters and levels for the attached camera. To determine proper settings for these values reference specific camera data. These parameters should be checked first if a camera is not responding to commands. Sending "0" as the baud rate will disable use of the camera serial port. Setting the Query bit will instruct the PTCR to return the current values without changing them. The microcontroller will be reloaded with the new values once the parameters are stored. **These parameters should be properly configured before physically connecting the cameras for the first time or changing camera serial port levels.**

Currently, **only 8 bits, no parity may be set and any other setting will be ignored.** However, hooks have been left in to accommodate different formats in the future. The transmitted values will be ignored when simply querying the PTCR. Therefore, they do not need to be included.

Data	Format	Bytes	7	6	5	4	3	2	1	0
Command	60H	1								
Camera 1	xxH	1	Query	0	LVL[1]	0-3 Parity/Bits[2]		0-7 baud[3]		

[1]0 = RS-232/1 = RS-422
[2]0 = 8/N, 1 = 7/N, 2 = 7/E, 3 = 7/O (Only 8/None currently supported)
[3]0 = Disable, 1 = 9.6, 2 = 14.4, 3 = 19.2, 4 = 28.8, 5 = 38.4, 6 = 57.6, 7 = 115.2

Response	Format	Bytes	7	6	5	4	3	2	1	0
Command	60H	1								
Camera 1	xxH	1	0	0	LVL	0-3 Parity/Bits		0-7 baud		

1.15.2 "Get/Set Lens Parameters" Command:

Proper setting of these values allow the microcontroller to know how the zoom and focus motors should be powered, if they are installed, and if the pot reading needs to be inverted. Setting the correct minimum speed value will keep the zoom and focus motors from stalling. Maximum speed is always considered the full speed available from the motor driver. This parameter should be checked first if the zoom and focus functions are not working properly.

Setting the Query bit will instruct the PTCR to return the current value without changing it. Clearing the Query bit will actually load the new value into the EEPROM. The secondary microcontrollers will be loaded with the new values once the parameters are stored.

Data	Format	Bytes	7	6	5	4	3	2	1	0
Command	61H	1								
Lens	Bitset	1	Query	[1]ZE	[2]ZR	[3]ZI	[4]FE	[5]FR	[6]FI	[7]DRV
Lens Z Min	xxH	1	0-255 (00H-FFH) minimum zoom speed							
Lens F Min	xxH	1	0-255 (00H-FFH) minimum focus speed							

[1]1 = Enable the Zoom Function
[2]1 = Reverse the Zoom Motor's Normal Operation
[3]1 = Invert the Zoom Resolver Reading
[4]1 = Enable the Focus Function
[5]1 = Reverse the Focus Motor's Normal Operation
[6]1 = Invert the Focus Resolver Reading
[7]0 = +12VDC Bipolar Drive/1 = ±6VDC Crosspolar Drive (Canon)

Response	Format	Bytes	7	6	5	4	3	2	1	0
Command	61H	1								
Lens 1	Bitset	1	0	ZE	ZR	ZI	FE	FR	FI	DRV
Lens 1 Z Min	xxH	1	0-255 (00H-FFH) minimum zoom speed							
Lens 1 F Min	xxH	1	0-255 (00H-FFH) minimum focus speed							

1.15.3 "Command Camera" Command:

This command acts as a transfer wrapper around a camera-specific command. In order to use this command the user should build an entire camera command string including any extra bytes such as control and framing characters and checksums the camera needs for communications. The PTCR-required STX, identity and command number should be prepended to the front and the ETX and LRC should be appended to the end. If the standard method of transmitting other commands is used (ESC insertion) any conflicting values in the camera string will be automatically converted to "safe" values for transmission to the PTCR.

The PTCR will strip the camera command string out of this command, return any altered control character values to their original value, and transfer the string to the camera. Note that the camera command string can be variable length **but cannot exceed 24 bytes**. The PTCR will respond to this command, indicating that it was received correctly. Any data the camera has to return will be sent through the "Get Status" 31H command. Reference it for more information on receiving returned data from the camera.

Data	Format	Bytes	7	6	5	4	3	2	1	0
Command	62H	1								
Cam Count	xxH	1	Number of Byte in Command String to Follow							
Cam Cmd	xxH	1-24	Complete Camera Command String							

Response	Format	Bytes	7	6	5	4	3	2	1	0
Command	62H	1								

1.15.4 "Set/Get Camera Response Timeout" Command:

As stated above, once a byte of data is received from the camera, the PTCR will wait for a defined time for additional bytes to arrive before flagging the main processor to gather and return them. The timer will be restarted every time an additional byte arrives. This method will reduce the number of partial string returns and parsing repeats for the host. However, different cameras will require differing amounts of time to complete their responses. This command allows adjusting the timeout/transfer data timer in multiples of 4ms.

Data	Format	Bytes	7	6	5	4	3	2	1	0
Command	65H	1								
Timeout	xxH	1	Query	1 – 100 (* 4ms)						

Response	Format	Bytes	7	6	5	4	3	2	1	0
Command	65H	1								
Timeout	xxH	1	0	1 – 100 (* 4ms)						

1.15.5 "Get/Set Aux Control Outputs" Command

The camera interface connector provides two auxiliary control outputs. These may be used to activate wipers, lens washers, lights, etc. Each line provides a voltage set at the input voltage of the PTCR-20 and a switched return path. Maximum loading for these outputs is dictated by the capacity of the input power supply but should be restricted to 1.5 Amps each or less. The auxiliary lines are toggle on/toggle off and will remain in the mode set by the last 66H command. The lines will be off by default at power-up. Setting the Query bit will instruct the PTCR to return the current value without changing it. Clearing the Query bit will load the new value into the unit and activate the lines accordingly.

Data	Format	Bytes	7	6	5	4	3	2	1	0
Command	66H	1								
Aux Bits	xxH	1	Query	0	0	0	0	0	AUX2	AUX1

Response	Format	Bytes	7	6	5	4	3	2	1	0
Command	66H	1								
Aux Bits	xxH	1	0	0	0	0	0	0	AUX2	AUX1

[1]1 = Output On/0 = Output Off for AUX(Output Number)

1.15.6 "Get/Set Camera/Lens Type" Command

Cameras are split into two different families; those with external lenses and those with internal lenses. External lens zoom and focus position is typically directly controlled by the PTCR-20 lens motor drivers and position sensors. These drivers and sensors are configured using command 61H and operate in the same manner as the platform pan and tilt axes drives.

Cameras with internal lenses typically use a serial communications port for control of the zoom and focus functions. Unlike the generic motor drive system used with external lenses each camera has unique serial commands for zoom and focus control. The PTCR-20 must be made aware of the camera type being used in order to send and receive serial commands correctly. The camera types and associated values are listed below the command table.

Data	Format	Bytes	7	6	5	4	3	2	1	0
Command	67H	1								
Cam Type	xxH	1	Query	0	Camera Type (0 – 63)					

Response	Format	Bytes	7	6	5	4	3	2	1	0
Command	67H	1								
Cam Type	xxH	1	0	0	Camera Type (0 – 63)					

Camera Type
0 = External or No Lens Control, Command 61H takes precedence
1 = Sony FCB-EX480A
2 = Sony FCB-EX780

1.15.7 "Get/Set Auto Focus Mode" Command:

This command will allow the user to switch between manual and automatic focus mode on cameras supporting the mode and fitted with internal lenses. If auto focus mode is detected as set the jog focus and stored position focus commands from preset or tour execution will be ignored as auto focus will execute the focus move. This command will be ignored if the camera type is set to 0, i.e., external or no lens control. If the camera does not support auto focus mode or the camera type is 0 the returned bit will always be 0.

Setting the Query bit will instruct the PTCR to return the current value without changing it. Clearing the Query bit will actually load the new value. Note that the last setting will NOT be stored by the PTCR-20 but may be stored by the camera depending on camera type.

Data	Format	Bytes	7	6	5	4	3	2	1	0
Command	68H	1								
Lens	Bitset	1	Query	0	0	0	0	0	0	[1]AFE

[1]1 = Enable Auto Focus/0 = Disable Auto Focus

Response	Format	Bytes	7	6	5	4	3	2	1	0
Command	68H	1								
Lens 1	Bitset	1	0	0	0	0	0	0	0	AFE

1.16 PTCR Operating Parameters:

Most parameters may be queried from the PTCR. A reduced set of PTCR parameters may also be modified through the remote interface.

1.16.1 Setting Pan & Tilt Angle Corrections:

The pan & tilt unit is internally calibrated to reflect an absolute relationship between the bottom mounting plate, the enclosure, and the tilt frame. However, the situation may arise where the user wishes to offset the degree display. This may be a result of mounting orientation or the desired method of measuring. To correct the reading, the user may provide an offset value for the displayed pan and tilt coordinates. Entry of a positive offset will simply increase the respective degree display. Negative will decrease the degree display. Internal orientation will not change, only the displayed angle. Sometimes it is beneficial to correct coordinate display for the platform relative to your point of reference. This can be manually performed by calculating and entering pan and tilt angle offsets. However, the

"Align to Center" and "Align to Coordinate" commands listed below can be used to allow the PTCR to perform these calculations for you.

Note that any change in pan and tilt offset will also modify the displayed position of presets, soft limits, etc. The relative angles will be correct, however. For example, assume a unit has a 0° tilt offset, the tilt frame is level at 0° and the preset will move it to -20°. Executing the preset move will move the tilt frame to -20°. If a +10° tilt offset is loaded a unit with a level tilt frame will display +10° and, after moving to the preset, -10° will be displayed. The unit has still moved 20° relative to the unit. The displayed angle has been altered to accommodate for the offset. **Therefore, it is recommended that any modification of pan & tilt angle offset be followed by a reloading of the preset table and soft limits into the host if normally displayed in your application.**

1.16.2 "Get Pan & Tilt Angle Correction" Command:

Data	Format	Bytes
Command	85H	1

Response	Format	Bytes	
Command	85H	1	
Pan Offset	Int	2	PAN = -1800 to +1800 = -180.0° to +180.0°
Tilt Offset	Int	2	TILT = -900 to +900 = -90.0° to +90.0°

1.16.3 “Set Pan & Tilt Angle Correction” Command:

The PTCR will check the input range of both pan and tilt angle corrections and will not save invalid entries.

Data	Format	Bytes	
Command	80H	1	
Pan Offset	Int	2	PAN = -1800 to +1800 = -180.0° to +180.0°
Tilt Offset	Int	2	TILT = -900 to +900 = -90.0° to +90.0°

Response	Format	Bytes	
Command	80H	1	
Pan Offset	Int	2	PAN = -1800 to +1800 = -180.0° to +180.0°
Tilt Offset	Int	2	TILT = -900 to +900 = -90.0° to +90.0°

1.16.4 “Align To Center” Command:

"Align To Center" will automatically calculate the pan and tilt angle corrections required to realign the angular position display for the platform so that the current position is considered a center position displaying a pan and tilt angle of 0°.

Data	Format	Bytes
Command	82H	1

Response	Format	Bytes	
Command	82H	1	
Pan Offset	Int	2	PAN = -1800 to +1800 = -180.0° to +180.0°
Tilt Offset	Int	2	TILT = -900 to +900 = -90.0° to +90.0°

1.16.5 “Align To Coordinate” Command:

"Align To Coordinate" allows entry of the desired position to display. For example, the platform is mounted on a southeast line, is jogged due east, and currently reads -45.0° in pan. The user wants this reading to be +90.0° reflecting a compass point. The user may calculate and manually enter an offset using Command 80H. Alternately, the user can simply enter +90.0° for pan using this command and the offset will be automatically calculated and stored.

Data	Format	Bytes	
Command	83H	1	
PAN Coord	Int	2	PAN = -1800 to +1800 = -180.0° to +180.0°
TILT Coord	Int	2	TILT = -900 to +900 = -90.0° to +90.0°

Response	Format	Bytes	
Command	83H	1	
Pan Offset	Int	2	PAN = -1800 to +1800 = -180.0° to +180.0°
Tilt Offset	Int	2	TILT = -900 to +900 = -90.0° to +90.0°

1.16.6 “Clear Angle Corrections” Command:

This command will reset any angular corrections to zero, realigning the platform angular display to the true 0/0 position.

Data	Format	Bytes
Command	84H	1

Response	Format	Bytes	
Command	84H	1	
Pan Offset	Int	2	PAN = 0
Tilt Offset	Int	2	TILT = 0

1.16.7 "Get/Set Pan & Tilt Soft Limits" Command:

The PTCR can contain degree positions that, when exceeded, can stop platform travel. These values are referred to as software or soft limits. Soft limits act as redundant safety stops in addition to the hard limit switches. Soft limits are normally set just inside the hard limits, making the soft limit the primary stop and the hard limit the redundant stop.

Though it would be possible to allow setting the soft limits by entering coordinates through the remote interface this feature has not been included in the interests of safety. We feel it is critical that the platform be observed while soft limits are being set in order to avoid collisions. Therefore, soft limits can only be set using a "move to and assign" method.

The user should jog the platform to the desired limit position, then send the command with the appropriate axis identified in order to set the soft limit. The user may override any existing soft limit by setting the OSL (Override Soft Limit) bit in the jog command. **This bit should only be used to assist in establishing soft limits.** The returned OSLR will show when this bit is set.

Note that any change in pan and tilt offset will also modify the displayed position of presets, soft limits, etc. The relative angles will be correct, however. Therefore, it is recommended that any modification of pan & tilt angle offset be followed by a reloading of the preset table and soft limits if normally displayed in your application.

1.16.8 "Get/Set Pan & Tilt Soft Limits" Command:

Data	Format	Bytes		
Command	81H	1		
Axis Number	xxH	1	Query	0 = CW, 1 = CCW, 2 = Up, 3 = Down

Response	Format	Bytes		
Command	81H	1		
Axis Number	xxH	1	0	0 = CW, 1 = CCW, 2 = Up, 3 = Down
Soft Limit	Int	2	PAN = -3600 to +3600, TILT = -1800 to +1800	

1.17 PTCR Factory Setup Parameters:

1.17.1 Getting/Setting Pan & Tilt Center Position:

The pan & tilt unit uses precision potentiometers or incremental encoders to track absolute position. A 12-bit ADC converts a voltage returned from each potentiometer to a digital value. This voltage is oversampled to create a psuedo 13-bit value. Therefore, the possible range of values is 0-8191. Each potentiometer should be centered as close as possible when the pan & tilt unit is at its midpoint of travel (tilt level, pan centered.) This will result in the best accuracy and reduce the chance of potentiometer damage due to over-travel.

On continuous rotation units the pan position feedback is provided by a 13-bit per revolution incremental encoder. Therefore, the possible range is also 0-8191 counts. A running count is maintained, incremented, or decremented every 200us according to the current state of the encoder. This value is then stored in non-volatile RAM. The encoder also provides an index pulse once per revolution. When initially configured at the factory, this index point is equated to a resolver value relative to absolute center in pan and is stored in non-volatile EEPROM (See Command 9DH.) The current running value stored in non-volatile RAM is updated frequently and will provide high accuracy at power up. However, if the platform is physically moved or bumped off position when powered down, the encoder changes will be missed and current position accuracy will be compromised. Whenever the index point is crossed during movement the encoder count will be automatically corrected using the stored index value. Optionally, the user may also reset to maximum accuracy by performing a homing cycle (See Command 9EH.) This command will automatically move the platform to the index point, realign the encoder count, and then return the platform to its original but corrected position. The center position for pan on a continuous rotation unit will always be 4096.

A yellow indicator for each axis will illuminate when the potentiometer or encoder reading is within ±24 counts (approximately ± 2°) of the center of travel of 4096. When initially aligning the unit the potentiometer sprockets should be loosened, the pan & tilt unit should be moved to a center position using jog (with soft limit override as required), the potentiometers should be adjusted until each yellow indicator illuminates, then the sprockets should be tightened. The user should then command the PTCR to save the potentiometer center positions. It is not critical that the potentiometers be exactly centered. The PTCR is able to discount any offset. It is important that the potentiometers be near center in order to reduce the possibility of over-travel damage. Note that the encoder is continuous rotation so over-travel is not a concern.

The first command will allow the user to retrieve the current stored value for potentiometer center for each axis in resolver units. A properly aligned system will return values very close to 4096.

The second command will allow the user to command the PTCR to read the potentiometers, then save the reading as the center positions. This should be performed immediately after the potentiometers have been aligned to the platform. The result will be re-aligned returned coordinates of 0°/0° (with companion 0°/0° angle offsets.)

1.17.2 "Get Pan & Tilt Potentiometer Center Position" Command:

Data	Format	Bytes
Command	90H	1

Response	Format	Bytes	
Command	90H	1	
Pan Count	Int	2	PAN = 0 – 8191
Tilt Count	Int	2	TILT = 0 – 8191

1.17.3 "Set Pan & Tilt Potentiometer Center Position" Command:

Data	Format	Bytes
Command	91H	1

Response	Format	Bytes	
Command	91H	1	
Pan Count	Int	2	PAN = 0 – 8191
Tilt Count	Int	2	TILT = 0 – 8191

1.17.4 "Get Minimum Speeds" Command:

The following two procedures allow the user to retrieve and set the minimum motor speeds for pan and tilt. Nominal settings will be stored in the microcontroller for each type of QPT. However, if the user is operating with an extremely heavy load or has finer positioning requirements with a very light load, jog and automated move functions may be more responsive when minimum speed is adjusted. The values of 0-255 represent total motor speed range from no power applied to absolute top speed.

The platform will not jog at less than minimum speed and will ramp down to minimum speed when completing an automated move. If numerous TO faults are detected during automated moves the user should increase the minimum speed for the faulted axis. The best way to determine the correct minimum speed is to install the load on the platform then attempt to jog the platform at low speed. The value should be adjusted until the platform moves as slowly as possible but no longer stalls. Tilt minimum speed should be set with the load at its "worst case" position, i.e., tilted to an extreme angle and jogged against the load (down and moving up or up and moving down.)

Data	Format	Bytes
Command	92H	1

Response	Format	Bytes	
Command	92H	1	
Pan Speed	xxH	1	Speed = 0 - 255
Tilt Speed	xxH	1	Speed = 0 - 255

1.17.5 "Set Minimum Speeds" Command:

Data	Format	Bytes	
Command	93H	1	
Pan Speed	xxH	1	Speed = 0 - 255
Tilt Speed	xxH	1	Speed = 0 - 255

Response	Format	Bytes	
Command	93H	1	
Pan Speed	xxH	1	Speed = 0 - 255
Tilt Speed	xxH	1	Speed = 0 - 255

1.17.6 "Initialize Preset Table to 0/0" Command:

This command initialize the entire preset table to 0°/0° (pan and tilt center) and set the zoom/focus bytes to 0. This can be used at initial setup to quickly clear the entire preset table. Reload the table into your application as required after issuing this command. **Actual reinitialization can take up to 3 seconds to complete before the PTCR responds.**

Data	Format	Bytes
Command	94H	1

Response	Format	Bytes
Command	94H	1

1.17.7 "Get/Set Motor and Potentiometer/Resolver Direction" Command:

As the controller is fitted to different QPT types, motors and resolvers will be connected through different drive systems that may require reversing readings or direction of rotation in order to make them relate properly to the platform. This command allows altering these configurations.

A device set for reverse operation is not necessarily an indication of miswiring or incorrect installation. It may simply be that the particular QPT type requires a motor or resolver to operate in the opposite direction due to design. For example, a non-continuous pan unit fitted with a potentiometer may increase its voltage output as the platform moves CW. A continuous rotation unit fitted with an encoder may actually decrease its count while moving clockwise. Both are correct. However, both will also return different directions of change when moving in the same physical direction. Configuring one unit as "Normal" and the other as "Inverted" in pan will correct this difference. A unit fitted with a higher or lower speed motor/gearbox combination may rotate the output shaft in the opposite direction. Therefore, though "Normal" mode may be the standard, this unit may operate in "Reverse" mode.

Setting a bit to 0 will configure that device for "Normal" mode. Setting the bit to "1" will configure it for "Reverse/Inverse" mode. These values are set at the factory and should not normally be altered.

Data	Format	Bytes	7	6	5	4	3	2	1	0
Command	95H	1								
Pan/Tilt	xxH	1	0	0	0	Query	TRES	PRES	TMTR	PMTR

Response	Format	Bytes	7	6	5	4	3	2	1	0
Command	95H	1								
Pan/Tilt	xxH	1	0	0	0	0	TRES	PRES	TMTR	PMTR

1.17.8 "Get Maximum Speeds" Command:

The following two procedures allow the user to retrieve and set the maximum motor speeds for pan and tilt units. If the user is operating with an extremely heavy load, jog and automated move functions may be more reliable when maximum speeds are adjusted. Adjustment of this value may also be worthwhile if the user wishes to perform an automated move at a specific speed (plotting of antenna patterns, time-intensive image recognition, etc.) The values of 1-255 represent a motor speed range from minimum to absolute top speed. If the maximum speed is set to less than minimum speed (see command 92H/93H) the platform will run no slower than minimum speed.

Data	Format	Bytes
Command	98H	1

Response	Format	Bytes	
Command	98H	1	
Pan Speed	xxH	1	Speed = 1 - 255
Tilt Speed	xxH	1	Speed = 1 - 255

1.17.9 "Set Maximum Speeds" Command:

Data	Format	Bytes	
Command	99H	1	
Pan Speed	xxH	1	Speed = 1 - 255
Tilt Speed	xxH	1	Speed = 1 - 255

Response	Format	Bytes	
Command	99H	1	
Pan Speed	xxH	1	Speed = 1 - 255
Tilt Speed	xxH	1	Speed = 1 - 255

1.17.10 "Get/Set Communication Timeout" Command:

The embedded controller's sole method of operating and providing feedback is via the communication interface. Timely return of position and status information is important for many applications. However, some applications exist where feedback is of little or no use. For example, if a QPT unit is used to simply move a camera through a series of presets for viewing, constant return of positional information may not be needed. The user is only interested in the picture returned by the camera. In this case, the user may wish to simply load a tour into the unit, start execution of the tour, then remove the communication connection and allow the unit to "free run." When operating in RS-485 mode it may not be possible or desired for a user's communications software to address all of the units in the communications daisy chain quickly. The capability to adjust or defeat the communication timeout value allows the user to assign a priority to the importance of constant communication.

Normally, a communication timeout is considered a fault of sufficient weight to stop any automated movement of the platform. If the data returned to the user's computer is critical, especially in determining the next move, the timeout should be set to a fairly low level (1-2 seconds.) If the user's software must share processing time and cannot service the QPT unit quickly a higher level can be set. If the user wishes the QPT to operate autonomously without the requirement for constant communication the user can set the timeout value for 0, defeating any stop due to a communication fault. Of course, all other faults will still remain active. Setting the Query bit will instruct the PTCR to return the current value without changing it. Clearing the Query bit will actually load the new value into the EEPROM and update the fault timer.

Data	Format	Bytes	7	6	5	4	3	2	1	0
Command	96H	1								
Timeout	xxH	1	Query	0(defeat) - 120 seconds						

Response	Format	Bytes	7	6	5	4	3	2	1	0
Command	96H	1								
Timeout	xxH	1	0	0(defeat) - 120 seconds						

1.17.11 "Get Firmware Revision" Command:

The revision D PTCR-20 introduced a microcontroller that can be reprogrammed in-circuit. Therefore, it is unrealistic to label the microcontroller with a firmware version. This command will allow the user to retrieve the revision number and date of the firmware installed in the PTCR-20.

Data	Format	Bytes	7	6	5	4	3	2	1	0
Command	9AH	1								

Response	Format	Bytes	7	6	5	4	3	2	1	0
Command	9AH	1								
Major Rev	xxH	1	0 – 255							
Minor Rev	xxH	1	0 – 255							
Date	xxH	1	1 – 31							
Month	xxH	1	1 – 12							
Year	xxH	1	0 – 99 (offset from 2000)							

1.17.12 "Get Clock" Command:

A realtime clock is included on the PTCR for providing both time and date and non-volatile storage of encoder position during power down. These two procedures provide a method of setting the clock and retrieving the current time and date. The clock is always based on 24 hours.

Data	Format	Bytes	7	6	5	4	3	2	1	0
Command	9BH	1								

Response	Format	Bytes	7	6	5	4	3	2	1	0
Command	9BH	1								
Seconds	xxH	1	0 - 59							
Minutes	xxH	1	0 - 59							
Hours	xxH	1	0 - 23							
Date	xxH	1	1 - 31							
Month	xxH	1	1 - 12							
Year	xxH	1	0 – 99 (offset from 2000)							

1.17.13 "Set Clock" Command:

Data	Format	Bytes	7	6	5	4	3	2	1	0
Command	9CH	1								
Seconds	xxH	1	0 – 59							
Minutes	xxH	1	0 - 59							
Hours	xxH	1	0 - 23							
Date	xxH	1	1 - 31							
Month	xxH	1	1 - 12							
Year	xxH	1	0 – 99 (offset from 2000)							

Response	Format	Bytes	7	6	5	4	3	2	1	0
Command	9CH	1								

Seconds	xxH	1	0 - 59
Minutes	xxH	1	0 - 59
Hours	xxH	1	0 - 23
Date	xxH	1	1 - 31
Month	xxH	1	1 - 12
Year	xxH	1	0 – 99 (offset from 2000)

1.17.14 "Initial Encoder Align" Command:

Continuous rotation units use an incremental encoder for pan position sensing. As the encoder is incremental, not absolute, position is derived by calculating an angle from an encoder count carried in SRAM and stored in non-volatile RAM memory. While the unit is powered up the SRAM and NVRAM values will stay accurate but, if the unit loses power during a high speed move or is bumped when powered down, it is possible that some accuracy may be lost due to undetected, unstored transitions of the encoder.

Along with quadrature outputs, the encoder also outputs an indexing pulse at one position in the unit's full rotation. Detection of this indexing pulse helps the platform update and maintain accuracy. An offset is stored in EEPROM that shows the encoder's count offset from physical center. Whenever this index pulse is crossed and detected, the PTCR will reload this offset into the encoder count, restoring full accuracy.

This offset is determined using the "Initial Encoder Align" command. The user should physically align the platform by jogging to pan center. This command should then be issued. The platform will automatically move, looking for the indexing pulse. Once found, the offset from physical center will be calculated and the count will be stored.

When first powered up, the platform should be within a few counts of the correct reading, depending on how the unit was handled during the power down period. As the unit is operated and the index pulse is crossed the stored count offset will be automatically loaded, increasing the reading to full accuracy. Optionally, the user may also force a realignment to the index pulse by executing a homing cycle (see Command 9EH.) The platform will automatically move to the area where the index pulse is expected to occur, will update the count, then return to the corrected original position.

Data	Format	Bytes	7	6	5	4	3	2	1	0
Command	9DH	1								

Response	Format	Bytes	7	6	5	4	3	2	1	0
Command	9DH	1								
Offset	Int	2	0 - 8191							

1.17.15 "Perform Homing Cycle" Command

Continuous rotation QPT-20 units are fitted with an incremental encoder to track pan position. The encoder is periodically sampled and the current count/position is stored in non-volatile RAM. Along with quadrature output, the encoder provides an index pulse once per revolution. During initial alignment this position is equated to an offset count from center (see Command 9DH.) This absolute count is stored in EEPROM memory. Whenever the pan movement of the platform causes the encoder to cross the index point the internal encoder position counter will be reset to this value to insure greatest accuracy. However, if the platform is treated roughly when powered down or loses power during high speed movement it is possible that the encoder count may become inaccurate due to missed transitions from the encoder. The homing cycle command allows forcing the platform to automatically hunt for the index pulse and refresh the encoder count. When commanded, the platform will begin automatic pan movement at high speed searching for the index pulse. Once found, the platform will return to its original but corrected position.

Data	Format	Bytes	7	6	5	4	3	2	1	0
Command	9EH	1								

Response	Format	Bytes	7	6	5	4	3	2	1	0
Command	9EH	1								
Index Pos	Int	2	0 - 8191							
Return Pos	Int	2	0 - 8191							

1.17.16 "Get/Set Identity Address" Command:

The identity address is used to uniquely identify a unit in a daisy-chained RS-485 environment. When the identity address is sent only the unit with a matching identity address will parse the incoming data, execute the command, seize the host receive line, respond, then release the line. The absence of responses from the other units allows a clear path for data return on the receive data line.

This command can be used to initially set or change the unit's identity address. If the user knows the current address of the unit to modify the current address should be sent as the Identity with the new address sent as the New Identity data byte. The unit will respond with the current address as the Identity and the new address as the New Identity response byte. **From this point forward the unit's identity address has been changed and it will only respond to the new address.**

What if the user loses the identity address of a unit? It can still be retrieved or changed by sending the 00 "broadcast" as the identity address. Any command will result in the return of the current identity in the response. Note, however, that any and all units in the network will act upon and respond to a broadcast command. **Therefore, the unit to modify must be isolated from the other units by either disconnecting the other units from the daisy chain or by connecting the unit to modify directly to a dedicated host.**

When operating in a dedicated RS-232 or RS-422 mode the identity should be set for 00. This will cause the dedicated unit to seize and hold the host's receive line full time.

Data	Format	Bytes	7	6	5	4	3	2	1	0
Command	9FH	1								
New Identity	xxH	1	Query	1-99 (01H-63H) for RS-485 identity or 00 for dedicated						

Response	Format	Bytes	7	6	5	4	3	2	1	0
Command	9FH	1								
New Identity	xxH	1	0	New Identity = 00-99 (00H-63H)						

1.17.17 "Get/Set Ramp Down Points in RU's" Command:

During automatic moves the platform runs at full speed but shifts down to minimum speed prior to reaching the destination in order to maintain accuracy. The distance from final destination that the shift takes place is referred to as the "ramp down point." For highest accuracy, the user should first set the minimum speed for an axis at the lowest speed possible without stalling, then adjust the ramp down point for each axis to achieve highest accuracy with a minimum of slow speed run time. The ramp down value is in resolver units.

Data	Format	Bytes	7	6	5	4	3	2	1	0
Command	86H	1								
Pan Ramp	xxH	1	Query	1-127 RU's						
Tilt Ramp	xxH	1	0	1-127 RU's						

Response	Format	Bytes	7	6	5	4	3	2	1	0
Command	86H	1								
Pan Ramp	xxH	1	Query	1-127 RU's						
Tilt Ramp	xxH	1	0	1-127 RU's						

2. Code Examples

The following snippets from the PTCR microcontroller code provide some examples as to how to implement efficient communications with the PTCR. Though the code is specifically written for a microcontroller compiler the unfamiliar commands should be self-explanatory. The snippets include LRC calculation and ESC/Bit-7 Set handling. Both transmit and receive procedures are interrupt based in these examples. **Note that, for the host's end, the handling of STX and ACK should be reversed.**

```
// Serial Communications Constants
#define  STX         0x02
#define  ETX         0x03
#define  ACK         0x06
#define  NAK         0x15
#define  ESC         0x1B


// Serial Transmit Interrupt Triggered on TX Buffer Empty
#int_tbe
tbe_isr() {
   if( tx_ptr == tx_len ) {
      putc( ETX );                        // send ETX and
      disable_interrupts(INT_TBE);        // buffer empty so disable transmit
   }
   else {
      switch( tx_buff[tx_ptr] ) {         // if a control character
         case STX :
         case ETX :
         case ESC :
         case ACK :
         case NAK :  putchar( ESC );                      // send an escape
                     bit_set( tx_buff[tx_ptr], 7 );       // and set bit 7 of next data
                     break;                               // byte in the buffer
         default  :  putchar( tx_buff[tx_ptr] );          // else just handle normally
                     tx_ptr++;
                     break;
      }
   }
}


// Serial Receive Interrupt
// Reception of an ETX must also follow setting an "STX found" flag so we
// will not try to parse a partial reception of data.
#int_rda
rda_isr() {
   int   temp_rx;
   temp_rx = getc();                      // get character from serial port
   switch( temp_rx ) {
      case STX :  rx_ptr = 0;             // realign to front of buffer
                  rx_done = FALSE;        // new stream coming, last now corrupt
                  esc_flag = FALSE;       // clear possible escape flag
                  found_id = FALSE;       // clear identity found flag
                  found_stx = TRUE;       // and indicate an STX was found
                  break;
      case ETX :  if( found_stx ) {       // if we started with an STX
                     rx_done = TRUE;      // flag done
```

```
                  found_stx = FALSE;    // clear to find another STX
               }
               found_id = FALSE;        // clear to find another identity
               esc_flag = FALSE;        // and clear possible escape flag
               break;
      case ESC : esc_flag = TRUE;       // escape found so flag
               break;
      default  : if( esc_flag ) {       // if last char was an escape
                  bit_clear( temp_rx, 7 );   // clear bit 7
                  esc_flag = FALSE;          // and clear flag
               }
               if(( found_STX ) && (!found_id)) {  next byte must be ID
                  rx_id = temp_rx;
                  found_id = TRUE;
               }
               else {
                  rx_buff[ rx_ptr ] = temp_rx;      // save byte
                  if( rx_ptr == 50 )                // prevents buffer overrun
                     rx_ptr = 0;
                  else
                     rx_ptr++;                      // and increment pointer
               }
               break;
   }
}

// Calculates and returns transmit LRC
// length is actually number of bytes, not top subscript
int Calc_LRC( int length ) {
   int temp_LRC = 0, index;
   for( index = 0; index < length; index++ )
      temp_LRC ^= tx_buff[index];
   return( temp_LRC );
}

// Calculates and returns TRUE if LRC was good or FALSE if LRC failed
short Check_LRC(void) {
   int temp_LRC, index;
   temp_LRC = rx_id;                    // prime with receive identity
   for( index = 0; index < rx_ptr; index++ )
      temp_LRC ^= rx_buff[index];
   return( temp_lrc == 0 );
}


// Send data
void send_data( int cmd_num ) {
   tx_buff[0] = unit_id;
   tx_buff[1] = cmd_num;
   tx_buff[2] = somedata;
   tx_buff[3] = moredata;
   tx_buff[4] = Calc_LRC( 4 );
   tx_len = 5;
   tx_ptr = 0;
   putchar(ACK);
   enable_interrupts(INT_TBE);       // triggers on TX buffer empty
```

```
}

void main( void ) {

disable_interrupts(INT_TBE);
enable_interrupts(INT_RDA);

< Code >

if( rx_done ) {                       // if we have received a packet
   rx_done = FALSE;                   // reset for next packet whether good or bad
   if( Check_LRC()) {                 // check the received LRC
      process data;                   // if good, process the data
      act_on_data;
      gather_return_data;
      send_data( echo_command );      // and send the response
   }
}
```

3. Revision History:

3.1 Rev A 12/22/01

Initial Release

3.2 Rev B 01/24/02 – 02/11/02

Entire camera communications system has been revamped. The user will wrap a complete camera command string in a pan & tilt command wrapper (see command 62H.) This will be sent to the pan & tilt. Since the protocol requires the ESC/bit-7 set method of passing any value that matches a control character, values embedded in the string that also match control characters will be properly sent and parsed on the distant end. The master microcontroller will remove the intact string from the packet and transfer it to the slave microcontroller. The slave microcontroller UART will be configured for the camera's communications format (see command 60H.) The slave will take the packet verbatim and send it to the camera via the UART using these parameters.

When the camera has data to return it will send it to the slave microcontroller via the serial port. When a data byte is received the slave will place it in a buffer and start a multi-millisecond timeout sequence (see command 65H.) If another byte arrives from the camera before timeout it will add it to the buffer and restart the timeout. Once the timeout expires the slave will signal the master and pass the packet. The master will queue the string and return it to the computer during the next "Get Status" query (see command 31H.) A byte will always be returned showing the length of this string. If this byte is 0 there is no camera data for the computer. If this byte is > 0 the computer should pull this string out of the packet and use it as required. The string will always be completely intact, consisting of all control characters, data bytes, and checksums in the proper sequence.

Increments for command 65H "Camera Timeout" is now 7ms rather than 10ms.

3.3 Rev C 10/10/02

The microcontroller for the PTCR-20 has been upgraded. Support for pan encoder interface has been added to provide continuous rotation capability. This required the addition of numerous commands to the protocol.

Cmd 41H Save Coordinates As Preset Table Entry

Removed transmission of absolute zoom and focus coordinates as these coordinates are fully relative. Current zoom/focus position can be added to the preset table entry by using either command 42H or 43H.

Cmd 43H Add Current Zoom/Focus To Preset Table Entry
Redefined from "Add Auxiliary Bytes to Preset Table Entry" to "Add Current Zoom/Focus To Preset Table Entry". This change is in description only. Its format has not changed.

Cmd 52H-55H Tour Manipulation
Maximum dwell time has been reduced from 255 seconds to 99 seconds.

Cmd 56H Get Tour Size
A new method of storing the tour has been implemented using a linked list. This command allows querying the list and determining the actual size of the tour. If a tour size of FFH is returned the tour has been corrupted.

Cmd 60H Get/Set Camera Comm Parameters
The PTCR has been modified to allow both RS-232 and RS-422 serial communications levels. This command has been modified to allow setting these levels. The values for baud rate have also been modified.

Cmd 61H Get/Set Lens Parameters
Restructured to provide a more generic method of configuration. Rather than specifying a lens type the user may configure the lens motor drivers and position feedback for normal/inverted operation, minimum speed, and +12VDC bipolar or ±6VDC crosspolar motor drive.

Cmd 63H Get/Set Camera Timeouts
Increments are now 4ms rather than 7ms.

Cmd 86H Get/Set Ramp Down Points in RU's
For greater accuracy across multiple speed ranges the maximum speed to minimum speed shift point (ramp down point) can now be adjusted using command 86H. The value is arbitrary and can be set by trial and error for best speed and accuracy.

Cmd 97H Get/Set Heater Power Sharing
As the QPT-20 Quickeye system contains a self-regulating heater this command has been removed.

Cmd 9AH Get Firmware Version
Since the microcontroller used can be reflashed with new code rather than replaced, labels for new microcontrollers reflecting updated code revision numbers can no longer be used. A firmware version value will be written into the code and can be retrieved using this command.

Cmd 9BH/9CH Get/Set Clock
A small non-volatile RAM with a realtime clock is included on the board for logging encoder travel. The realtime clock can be set and its contents retrieved by using these commands.

Cmd 9DH Initial Encoder Align
This command is used to initially align the encoder to a factory 0/0 position.

Cmd 9EH Perform Homing Cycle
This command is used to find the index position of encoder-based continuous rotation units in order to improve pan angle accuracy.

Cmd 9FH Get/Set Identity Address

Rotary DIP switches for addressing were removed from the PTCR-20 at Rev D. The address is now a non-volatile parameter stored in EEPROM.

3.4 Rev D 05/28/03

The command set has been updated to include both implementation of the auxiliary outputs and serial-based camera zoom and focus control.

Cmd 66H Get/Set Aux Control Outputs
Allows activation and query of the two auxiliary camera outputs.

Cmd 67H Get/Set Camera/Lens Type
Selects either the generic motor driven external lens interface or allows the PTCR-20 to communicate with specific cameras fitted with serial port control and internal lenses.

Cmd 68H Get/Set Auto Focus Mode
Enables or disables auto focus on specific cameras fitted with serial port control and internal lenses.